

This listing of claims will replace all prior versions, and listings, of claims in the application:

**Listing of Claims:**

1           1. (original) An intermediate object linking method of  
2 linking a plurality of intermediate objects to form an  
3 executable object, comprising:  
4           an intermediate object linking order forming step which  
5               decides linking orders of the plurality of  
6               intermediate objects;  
7           a linking processing step which executes linking  
8               processes of the plurality of intermediate objects  
9               based on the linking orders decided by the  
10              intermediate object linking order forming step to  
11              get an executable object;  
12           a comparing step which compares program size of the  
13               executable objects obtained by the linking  
14               processing step with the program size of a  
15               executable objects stored in a storing section every  
16               time when the linking order is changed;  
17           a storing step for storing the program size of the  
18               executable objects and the linking order obtained by  
19               the linking processing step in the storing section  
20               to update when the program size of the executable  
21               objects obtained by the linking processing step is  
22               smaller than the program size of the executable  
23               objects stored in the storing section at the  
24               comparing step; and  
25           a repeating step for changing the linking orders by the  
26               intermediate object linking order forming step and  
27               executing repeatedly the linking processing step,  
28               the comparing step, and the storing step.

1        2. (original) An intermediate object linking method  
2 according to claim 1, wherein the intermediate object linking  
3 order forming step decides the linking orders by a permutation  
4 algorithm.

1        3. (original) An intermediate object linking method  
2 according to claim 1, wherein the intermediate object linking  
3 order forming step decides the linking orders by a genetic  
4 algorithm.

1        4. (currently amended) An intermediate object linking  
2 unit for linking a plurality of intermediate objects to form  
3 an executable object, comprising:  
4        an intermediate object linking order forming section  
5                which decides linking orders of the plurality of  
6                intermediate objects;  
7        a linker starting section which executes linking  
8                processes of the plurality of intermediate objects  
9                based on the linking orders decided by the  
10                intermediate object linking order forming section to  
11                form an executable object;  
12        a comparing section which compares program size of the  
13                executable objects every time when the linking order  
14                is changed; a storing section which stores the  
15                program size of the executable objects and the  
16                linking order; and  
17        a repeating section which changes the linking orders by  
18                the intermediate object linking order forming  
19                section and  
20        operating repeatedly the linker starting section, the  
21                comparing section, and the storing section,  
22        wherein the comparing section compares the program size  
23                of the executable objects formed by the linker

24           starting section with the program size of the  
25           executable objects stored in the storing section,  
26       wherein the storing section stores the program size of  
27           the executable objects and the ~~linker~~ linking order  
28           formed by the linker starting section when the  
29           program size of the executable objects formed by the  
30           linker starting section is smaller than the program  
31           size of the executable objects stored in the storing  
32           section at the comparison by the comparing section.

1           5. (original) An intermediate object linking unit  
2       according to claim 4, wherein the intermediate object linking  
3       order forming section decides the linking orders by a  
4       permutation algorithm.

1           6. (original) An intermediate object linking unit  
2       according to claim 4, wherein the intermediate object linking  
3       order forming section decides the linking orders by a genetic  
4       algorithm.

1           7. (currently amended) A linker unit for linking a  
2       plurality of intermediate objects to form an executable  
3       object, comprising:  
4           an intermediate object linking order forming section  
5               which decides linking orders of the plurality of  
6               intermediate objects;  
7           a linking processing section which executes linking  
8               processes of the plurality of intermediate objects  
9               based on the linking orders decided by the  
10           intermediate object linking order forming section to  
11           form an executable object;  
12           a comparing section which compares program size of the  
13               executable objects every time when the linking order  
14           is changed; a storing section which stores the

15           program size of the executable objects and the  
16           linking order; and  
17       a repeating section which changes the linking orders by  
18           the intermediate object linking order forming  
19           section and operating repeatedly the linker starting  
20           section, the comparing section, and the storing  
21           section,  
22       wherein the comparing section compares the program size  
23           of the executable objects formed by the linker  
24           starting section with the program size of the  
25           executable objects stored in the storing section,  
26       wherein the storing section stores the program size of  
27           the executable objects and the ~~inking~~ linking order  
28           formed by the linker starting section when the  
29           program size of the executable objects formed by the  
30           linker starting section is smaller than the program  
31           size of the executable objects stored in the storing  
32           section at the comparison by the comparing section.

1       8. (original) A linker unit according to claim 7, wherein  
2       the intermediate object linking order forming section decides  
3       the linking orders by a permutation algorithm.

1       9. (original) A linker unit according to claim 7, wherein  
2       the intermediate object linking order forming section decides  
3       the linking orders by a genetic algorithm.

1       10. (original) A compiler driving unit for translating a  
2       source program by starting a compiler, an assembler, a linker,  
3       etc. to form an executable object, comprising:  
4       an intermediate object linking order forming section  
5           which decides linking orders of the plurality of  
6       intermediate objects;

7 a linker starting section which executes linking  
8 processes of the plurality of intermediate objects  
9 based on the linking orders decided by the  
10 intermediate object linking order forming section to  
11 form an executable object;  
12 a comparing section which compares program size of the  
13 executable objects every time when the linking order  
14 is changed; a storing section which stores the  
15 program size of the executable objects and the  
16 linking order; and  
17 a repeating section which changes the linking orders by  
18 the intermediate object linking order forming  
19 section and operating repeatedly the linker starting  
20 section, the comparing section, and the storing  
21 section,  
22 wherein the comparing section compares the program size  
23 of the executable objects formed by the linker  
24 starting section with the program size of the  
25 executable objects stored in the storing section,  
26 wherein the storing section stores the program size of  
27 the executable objects and the ~~inking~~ linking order  
28 formed by the linker starting section when the  
29 program size of the executable objects formed by the  
30 linker starting section is smaller than the program  
31 size of the executable objects stored in the storing  
32 section at the comparison by the comparing section.

1 11. (original) A compiler driving unit according to claim  
2 10, wherein the intermediate object linking order forming  
3 section decides the linking orders by a permutation algorithm.

1 12. (original) A compiler driving unit according to claim  
2 10, wherein the intermediate object linking order forming  
3 section decides the linking orders by a genetic algorithm.

1        13. (original) A recording medium for recording a program  
2 for linking a plurality of intermediate objects to form an  
3 executable object, wherein the program for causing a computer  
4 to execute a method comprising:

5        an intermediate object linking order forming step which  
6                decides linking orders of the plurality of  
7                intermediate objects;

8        a linking processing step which executes linking  
9                processes of the plurality of intermediate objects  
10               based on the linking orders decided by the  
11               intermediate object linking order forming step to  
12               get an executable object;

13       a comparing step which compares program size of the  
14               executable objects obtained by the linking  
15               processing step with the program size of a  
16               executable objects stored in a storing section every  
17               time when the linking order is changed;

18       a storing step for storing the program size of the  
19               executable objects and the linking order obtained by  
20               the linking processing step in the storing section  
21               to update when the program size of the executable  
22               objects obtained by the linking processing step is  
23               smaller, than the program size of the executable  
24               objects stored in the storing section at the  
25               comparing step; and

26       a repeating step for changing the linking orders by the  
27               intermediate object linking order forming step and  
28               executing repeatedly the linking processing step,  
29               the comparing step, and the storing step.

1        14. (original) A recording medium according to claim 13,  
2 wherein the intermediate object linking order forming step  
3 decides the linking orders by a permutation algorithm.

- 1           15. (original) A recording medium according to claim 13,
- 2 wherein the intermediate object linking order forming step
- 3 decides the linking orders by a genetic algorithm.